The STEP Data Integration Architecture Activity

David M. Price
IBM
5300 International Boulevard
North Charleston, South Carolina, USA
29418

Phone +1 843-760-4341
Fax +1 843-760-3349
dmprice@us.ibm.com

Based on work by:
Matthew West
Shell Services International
Shell Centre
London SE1 7NA, UK
+44 (0) 171 934 4490
matthew_west@compuserve.com

Julian Fowler
PDT Solutions
22 Greenwood Street
Altrincham WA14 1RZ, UK
+44 (0) 161 929 4437
jfowler@pdtsolutions.co.uk

Abstract

Standardized data models for products and process have been developed in the context of the ISO TC184/SC4 Industrial Data subcommittee, the primary product of which is ISO 10303, the STEP Standard for the Exchange of Product model data. The benefits of exchanging design engineering data between applications using STEP are now being realized. However, more data interoperability, higher levels of enterprise integration, and more convenient data sharing are needed. Product information expressed using metastructures other than STEP Application Protocols, as well as many kinds of information that are not product information, must be integrated and shared by multiple applications and users concurrently. The models of such information may or may not be defined using STEP's EXPRESS language.

Standards-based capabilities for exchange and integration of industrial data can be extended without compromising existing investments in STEP, PLIB, SGML, etc. A data integration architecture can support a broad set of requirements without replacing the underlying metastructures of existing data and applications.

Introduction

As an ever widening spectrum of industries have become involved in the development, implementation, and use of standards for industrial data, and have developed expectations of the results and benefits to be gained from the use of the SC4 standards, so a number of fundamental issues have been raised regarding the SC4 architectures and methodologies and their applicability to the broad set of requirements that industry brings to SC4. The most notable of these issues are [1]:

— interoperability of applications, i.e., the ability to communicate information amongst heterogeneous computer systems;

— co-operative use of STEP application protocols;

— co-operative use of the SC4 standards: STEP (ISO 10303 Product data representation and exchange), P-LIB (ISO 13584 Parts Library), MANDATE (ISO 15531 Industrial manufacturing management data), and Oil & Gas (ISO 15926 Integration of life-cycle data for oil and gas production facilities);

— co-operative use of the SC4 standards with other standards that overlap with the domain of “industrial data”, such as SGML, EDIFACT and those produced by OMG;

— integration of data, i.e., the management of data from diverse sources in an efficient and effective manner;

— development of “data sharing” implementations, i.e., of solutions that combine STEP data models with database management system technologies to facilitate concurrent engineering, life-cycle data management, etc.

This paper provides an overview of the proposed response to these issues that exploits work done in a number of the collaborative projects that contribute to the work of SC4. The scope of this response, the SC4 Data Integration Architecture activity, is the development of standard data specifications that meet industry requirements for data exchange (communication) and data sharing (integration).

Requirements and potential solutions

A fundamental set of requirements need to be satisfied to address the outlined issues [2]:

1. There is a requirement for a data model able to hold data from different sources independent of the usage of the data.
2. There is a requirement to be able to integrate product data with other industrial and business data.
3. In order to have the minimum number of data models for integrating data from different sources, these models are required to be extensible and modular.
4. There a requirement to create two way mappings between the application level data models and the integration level data models, for both the creation and exchange of data sets, and for being able to view and update the data directly.
5. There is a requirement to handle data models in languages other than EXPRESS.
6. There is a requirement to identify the commonality between different data models in a consistent and manageable manner.
7. There is a requirement to consolidate data sets that contain data about the same objects. In particular this means being able to define when you are satisfied two records are about the same object.

However, the primary requirement is to support the integration of industrial data from different data sets, according to different data models, into a single data set, and a single data model. The data in this model might then need to be viewed from a perspective defined by yet another data model. The ANSI/SPARC three layered architecture labelled a data model that could support the data for a number of data models as a conceptual data model. This term will be used in this paper to describe the relationship between one data model and another. Thus one data model might be conceptual relative to two or more others, i.e. it can support all of their data requirements (but not necessarily constraints).

While it is well understood how to develop a conceptual data model from some predefined set of external models, this could give rise to a large number of conceptual data models for the different combinations that arise. It would be desirable to have only one, or a few conceptual data models. However, since all the data models that need to be integrated will not be known at the start of the process, it is important that any conceptual data model is extensible in the face of additional information requirements. For our purposes this model must also be able to integrate product data models with data models of other parts of a business.

Development of such an extensible conceptual data model is to be the central deliverable of the SC4 data integration architecture activity. This type of data model has been termed a Generic Data Model and such a data model will:

- meet the data requirement,
- be clear and unambiguous to all (not just the authors),
- be stable in the face of changing data requirements,
- be flexible in the face of changing business practices,
- be reusable by others,
- be consistent with other generic data models covering the same scope, and
- be able to integrate data from different data models.

Fortunately, an initial set of principles exist which, if followed, allow these requirements to be met. They are:

1. Candidate attributes should be treated as representing relationships to other entity types.
2. Entities should have a local identifier within a database or exchange file. These should be artificial and managed to be unique. Relationships should not be used as part of the local identifier. (External, or global, identifiers are objects in their own right, see principle 1.)
3. Activities, associations and event-effects should be represented by entity types (not relationships or attributes).
4. Relationships (in the entity/relationship sense) should only be used to express the involvement of entity types with activities or associations.
5. Entity types should represent, and be named after, the underlying nature of an object, not the role it plays in a particular context.
6. Entity types should be part of a subtype/ super type hierarchy of generic entity types in order to define a universal context for the model.

Developing data models that follow these principles has been found to lead to data models that satisfy the above requirements. These are not yet the final set of principles that will guide our development but are the basis from which the data integration model has been developed.

Classification Schemes

When you develop a Generic Data Model much of the semantic information is taken out of the data model. If this is not to be lost then it needs to be held somewhere else. An appropriate place to hold this information is in populations of the Generic Data Model. These populations are to be classification schemes which also contains information about the relationships between classification schemes that represent our knowledge about some part of the universe. Two types of classification schemes are recognised:

1. Pure classification schemes, here the basis for classification is the same throughout the classification hierarchy. Such classification schemes can be

expected to be orthogonal. Here orthogonal means that membership of a class in one hierarchy is independent of membership of another hierarchy, and that classes at any level will be mutually exclusive.

2. Mixed classification schemes, these are the combinations of classes of thing we are interested in for a purpose. They are generally of more practical use, but can easily overlap with each other. The overlaps can be managed through the elements that make up the mixed classification schemes coming from pure classification schemes.

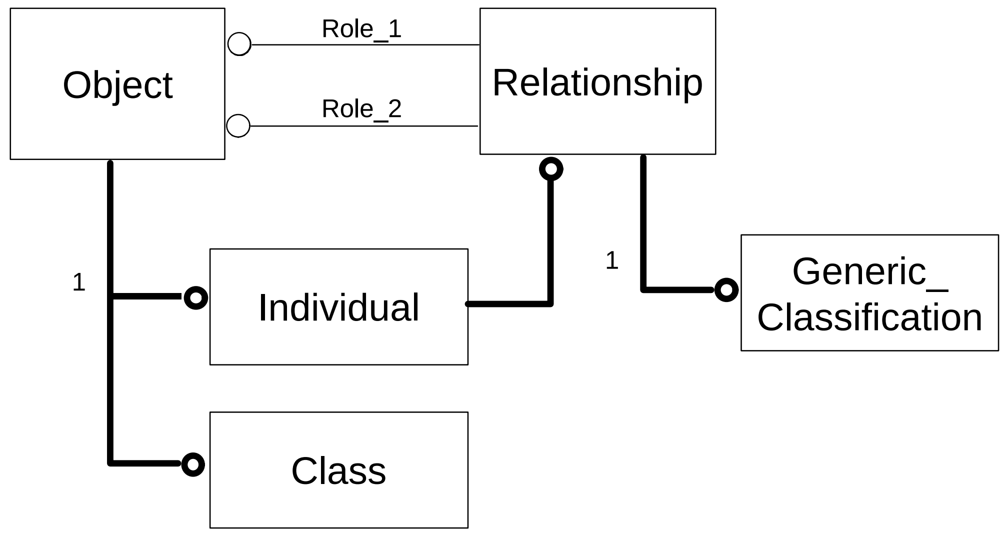
Pure classification schemes tend to be concise, whereas the mixed classification schemes that can be derived from them are relatively large. This means that pure classification schemes can be a useful tool for managing the mapping between data models. An entity type in a data model can be positioned (or not) within each of the pure classification schemes. Doing so defines very precisely the meaning of an entity type, in a way that is consistent. Thus if the pure classification schemes are considered to be an adjunct to the generic data model, then a consistent and extensible basis exists for integrating data from different data models. Further, if a data model is developed that can be driven by classification schemes, then the data model no longer needs to be extensible, only the classification schemes (using the same principles). Thus extensibility becomes a matter of managing additions to data rather than data models. These additions generally take the form of subclasses that are intersections between at least two more fundamental superclasses.

Alternatively, the classification schemes can be used to specialise the data model to make greater detail explicit, should this be required. It is important to understand that declaring something as a member of an entity type is semantically identical to classifying it as being a member of a class, where the definition of the class and the entity type are the same. They are just different representations/presentations of the same information.

Metastructural Foundations for Classification Schemes

By following the principles for creating generic data models and remaining mindful of the objectives for an integration architecture based on classification schemes, a set of metastructural foundations have been developed. They focus on looking at what things are - in particular trying to identify dependence of concepts. The basic concepts from which we begin are: object, class, individual, relationship and classification.

Object is anything that exists, real or imagined. It is the supertype, or superclass, of everything else. Class is a sets of objects that have a basis for inclusion or exclusion for membership. Class is a type of object. Individual is an object that is a particular thing, not a class. Individual is a type of object. Relationship is a binary tuple that indicates what one object has to do with another. Relationship is a type of individual. Classification indicates that an object is a member of a class. Classification is a type of relationship. A data model designed to be driven by classification schemes, and to hold any data about any thing, is given in Figure 1.



- dark lines identify supertype/superclass relationships
- the one (1) indicates that the subtypes/subclasses are mutually exclusive
- light lines identify attributes of types/classes

Figure 1 : Metastructures for an Ontology

Data Integration

Given the metastructures supporting a classification scheme, the next step is to support the creation of the ontology via the integration of industrial data from different data sets and data models. The development of this data integration process is not complete but several key aspects have been identified and are in work. These aspects are [2]:

- formal mapping between data models which involves two processes
  - it takes the (implicit) context of an existing data model and maps it to and from explicit elements in the generic, and more conceptual, data model;
  - it takes the explicit elements of an existing data model and maps them to and from equivalent elements in the generic, and more conceptual, data model.
- formal mapping between data models must be based on discovering the relative semantics of the models - and the actual usage of the model in cases where all uses do not conform to the semantics of the existing data model.

- identifying which data based on different data models pertains to the same object - and thus needs to be consolidated.
  - including identification within the integrated dataset
  - including external identification for computer systems, people and organizations outside the integrated dataset
- data consolidation so that data about the same object based on different data models is integrated in the dataset based on the conceptual data model
  - it is possible that situations will arise such that only human intervention can determine whether two objects in the existing dataset are about the same object in the integrated dataset

Supporting the Life Cycle

In addition to the ontological approach taken for the SC4 Data Integration Architecture a further strong requirement for supporting data over a long life cycle must be met. To address this requirement, several key generic portions of the ontology have already been developed. The most fundamental of these is the concept of an information content. Information content is a type of individual. It is what is used to describe an object. Information contents are classified and have a named, via another information content, relationship to the object they describe. The major requirement this concept is defined to meet is to enable systems to maintain different values for the same characteristic of an object over time. Example types of information content are text value and integer value.

External Data Models and Model Projection and Translation

As the ANSI/SPARC architecture made clear, conceptual data models and generic data integration models are not appropriate for all uses. To meet requirements for external and physical data models a data model projection and translation capability has been identified. Formalizing the projection of the integration data model and the generic portions of the ontology into "flatter" models supporting data exchange scenarios where data access and life cycle support are not required is the initial goal. A mapping language project is underway in SC4 and a language called EXPRESS-X is being developed. The two major requirements from the data integration architecture are to support the projection of models between different levels of abstraction and the translation of data elements where terminology differs between models at the same level of abstraction.

Progress and Issues

To date the SC4 Data Integration Architecture activity has documented the business requirements driving its development and identified the key technical requirements and many of the main architectural elements that we believe will meet the given requirements. The very generic, conceptual data model and its related fundamental concepts have been proposed and an initial set of real-world test cases mapping into the conceptual model are underway based on the other existing SC4 standards.

The majority of the development over the past year focused on the metastructures and ontology required for a very conceptual data integration architecture. Issues and requirements in many additional areas including the following have been identified:

- support for data sharing implementation methods
- support for maintaining the constraints found in application data models
- support for behaviour associated with objects
- language support for instances and classes and more capable relationships between the two
- a very disciplined integration process
- integration of data models defined using SGML and related standards

Figure 2 depicts the concepts developed to date and summarizes the data model architecture. Work in the other areas described above will become a higher priority once the fundamental concepts are accepted. The SC4 Data Integration Architecture activity is an ambitious project. However, incremental, timely and useful interim deliverables are expected in 1999 and 2000.

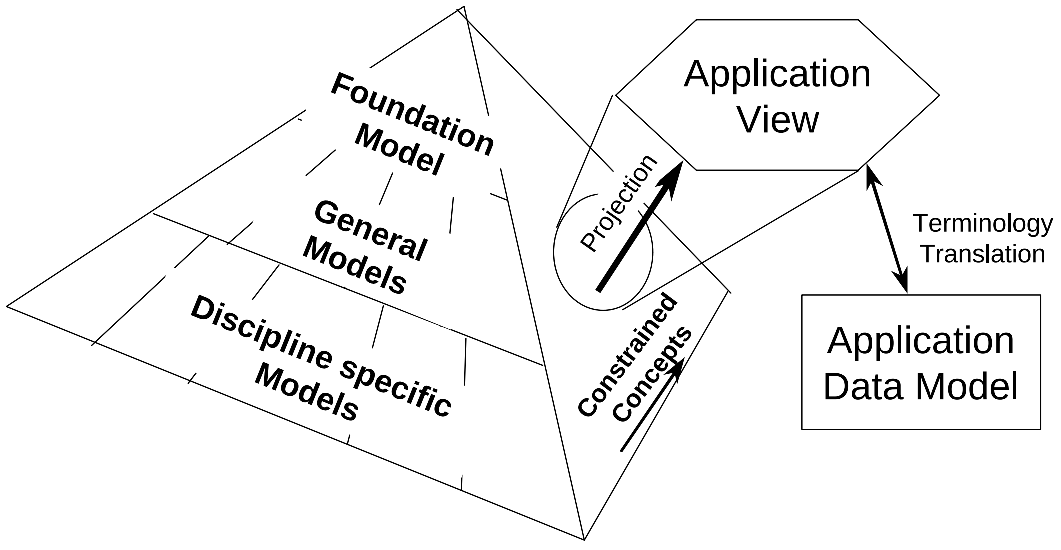


Figure 2 : A Data Integration Archtecture

References

[1] West, Matthew; Fowler, Julian. The Nature of Industrial Data. ISO TC184/SC4/WG10 N64, April 1996.

[2] West, Matthew. Integration of Industrial Data for Exchange, Access and Sharing (IIDEAS). ISO TC184/SC4/WG10 N85, December 1996.